SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni diplomski studij računarstva

BROJAČ OSOBA ZASNOVAN NA OBRADI SLIKE

Diplomski rad

Hrvoje Karalić

Osijek, 2018.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za obranu diplomskog rada

Osijek, 24.09.2018.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu diplomskog rada

Ime i prezime studenta:	Hrvoje Karalić
Studij, smjer:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D 698 R, 25.09.2017.
OIB studenta:	01505952191
Mentor:	Doc.dr.sc. Ratko Grbić
Sumentor:	
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	Doc.dr.sc. Emmanuel-Karlo Nyarko
Član Povjerenstva:	Petra Đurović
Naslov diplomskog rada:	Brojač osoba zasnovan na obradi slike
Znanstvena grana rada:	**Obradba informacija (zn. polje računarstvo)**
Zadatak diplomskog rada:	U okviru diplomskog rada potrebno je razviti PC aplikaciju za brojanje osoba koje ulaze/izlaze iz određenog prostora. Brojanje osoba se zasniva na obradi slike pomoću odgovarajućih algoritama pri čemu se slika dobiva preko web kamere. U okviru rada potrebno je ispitati mogućnosti brojanja u realnom vremenu i implementacije u Raspberry Pi jednokartično računalo. Sumentor: -
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Dobar (3)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 2 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 2 bod/boda Jasnoća pismenog izražavanja: 1 bod/boda Razina samostalnosti: 2 razina
Datum prijedloga ocjene mentora:	24.09.2018.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 03.10.2018.

Ime i prezime studenta:	Hrvoje Karalić
Studij:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D 698 R, 25.09.2017.
Ephorus podudaranje [%]:	7

Ovom izjavom izjavljujem da je rad pod nazivom**: Brojač osoba zasnovan na obradi slike**

izrađen pod vodstvom mentora Doc.dr.sc. Ratko Grbić

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Brojanje osoba ili drugih objekata u kretanju pri ulasku i izlasku iz prostorije ili preko nekog drugog zamišljenog praga otvara mogućnost prikupljanja čitavog niza informacija. Brojač osoba je aplikacija koji se sastoji od jednog ili više senzora koji u realnom vremenu bilježe koliko ljudi prođe ispred njih i u kojem smjeru.

Aplikacija brojanja osoba u kretanju ima široku primjenu i može poslužiti u gotovo svim slučajevima javnih prostorija, objekata, prometnica i drugih mjesta visoke frekvencije kretanja gdje je snimanje dozvoljeno. Za brojanje osoba ta mjesta mogu biti: trgovine, lanci prodavaonica, shopping-centri i galerije, javne i upravne zgrade, parkirna mjesta i garaže, banke itd. Brojač osoba daje uvid u podatke koji mogu biti korisni iz znanstveno istraživačkih razloga ali i podatke koji mogu unaprijediti poslovanje i biti komercijalno upotrebljivi. S tim podacima moguće je: bolje organizirati radnu snagu, prilagoditi radno vrijeme ovisno o prometu ljudi, utvrditi omjer stvarnih kupaca i posjetitelja, uvid u obrt novca po posjetitelju, utvrditi efikasnost reklamnih kampanja, prepoznati i nagraditi više efikasne podružnice i zaposlenike, uvidjeti trendove tokom duljeg razdoblja, dobiti pomoć u planiranju troškova najma ili održavanja.

Brojanje osoba najčešće se obavlja pomoću termalnih brojača, brojača zanovanih na tehnologiji bežične mreže (WiFi), brojača zasnovanih na infracrvenoj svjetlosti i vremenu vraćanja zrake i brojača zasnovanih na kameri i računalnom vidu.

Cilj ovog rada je razviti rješenje za brojanje osoba zasnovano na kameri kao senzoru i odgovarajućoj računalnoj platformi kao što je PC ili Raspberry Pi. Brojanje osoba je izvedeno osmišljavanjem dva algoritma temeljena na funkciji apsolutne razlike između okvira i funkciji izračunavanja optičkog toka slike. Nadalje, cilj rada je ispitati ponašanje programskog rješenja u raznim uvjetima uključujući: vanjske i unutarnje lokacije, širinu prolaza (vrata), razinu osvjetljenja i prisutnost sjene i kut postavljanja kamere.

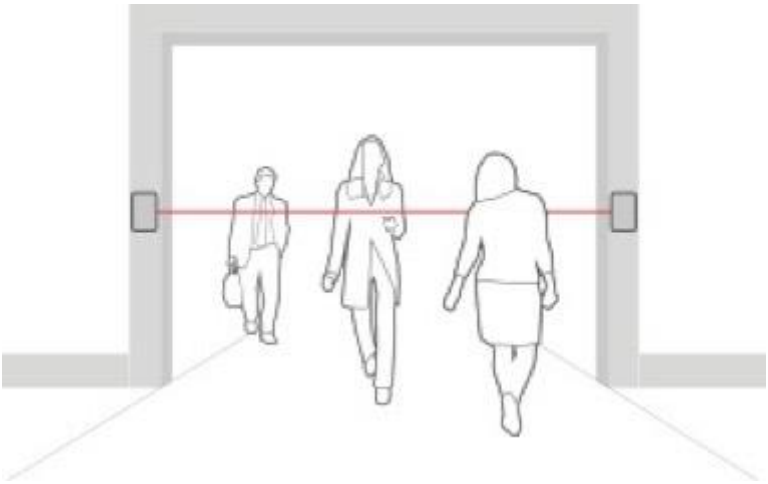
U prvom poglavlju ovog rada dan je uvod u problematiku brojača osoba, a u drugom upoznavanje s postojećim rješenjima zasnovanim na raznim tehnologijama. Treće poglavlje sadrži programski i sklopovski prijedlog rješenja opisan u radu, dok su u četvrtom rezultati provedenih mjerenja. Na kraju rada dan je zaključak.

2. METODE BROJANJA OSOBA

Prilikom upuštanja u razvoj sustava koji broji osobe istraživanjem nailazimo na razne metode i načine brojanja koji su se koristili ili se još uvijek koriste. Iako je povijest automatskog brojanja osoba uz pomoć tehnologije relativno kratka, postoji širola lepeza sklopovskih i programskih rješenja ovog problema. S razvojem tehnologije razvijaju se i generacije brojača osoba. U ovom poglavlju bit će opisana neka od rješenja koja su istražena za potrebe diplomskog rada.

2.1. Brojači zasnovani na infracrvenoj svjetlosti

Najjednostavniji oblik brojača ima jednu, horizontalnu infracrvenu zraku preko ulaza koja je obično povezana s malom LCD jedinicom-zaslonom. Takav sustav broji prekide u dolasku zrake na drugu stranu vrata pa je za ukupan broj osoba koje su pristupile u objekt potrebno podijeliti broj prekida s 2 jer svaka osoba izazove jedan prekid i pri ulasku i pri izlasku. Ova brojila obično zahtijevaju prijemnik ili reflektor montiran nasuprot jedinici koja emitira zraku s tipičnim rasponom od 2,5 metara do 6 metara. Prednost ovakvih brojača je u jednostavnosti implementacije i malim tehnoloških zahtjevima što znači manju cijenu. Nedostaci su primijećeni u točnosti brojanja budući da dvije ili više osoba mogu ući u prostoriju krećući se usporedno [1].



Sl.2.1. *Brojač osoba zasnovan na IC zrakama* [1]

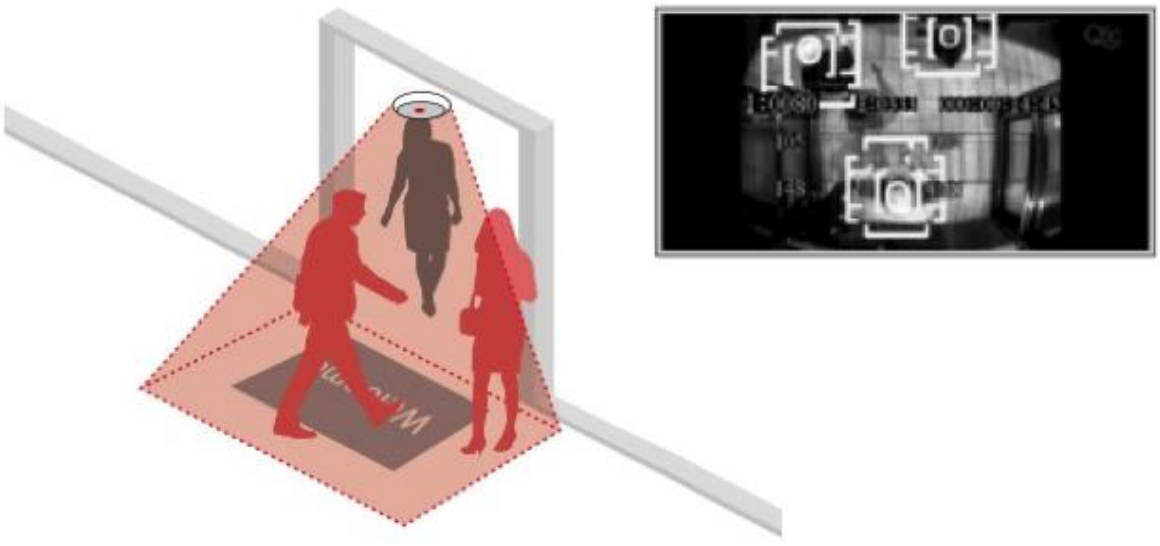
Na slici 2.1 vidimo ilustraciju brojača zasnovanog na infracrvenim zrakama. Osoba prilikom prolaska kroz vrata prekida snop infracrvene svjetlosti što prijemnik detektira i broji. Ovakva rješenja imaju relativno jeftino sklopovlje i programsku podršku.

2.2. Brojači osoba zasnovani na toplini

Toplinski sustavi za brojanje koriste senzore koji detektiraju zrake izvora topline. Ovi sustavi su obično postavljeni iznad glave osobe za visoke točnosti i otkrivaju izvore topline iz ljudskog tijela kao što je prikazano na slici 2.2.

Prije pojave naprednijih sustava, poput onih zasnovanih na kameri, termalni brojači bili su glavni izbor za većinu ustanova jer su prilično točni. Ipak, oni imaju svoja ograničenja:

- Toplinski brojači ne mogu se montirati na visokom stropu
- Mogu pokriti samo ulaz na uska vrata
- Teško je provjeriti točnost brojača
- Točnost je smanjena na mjestima s varijacijama u termalnim uvjetima [1]



Sl.2.2. *Brojač osoba zasnovan na toplini* [1]

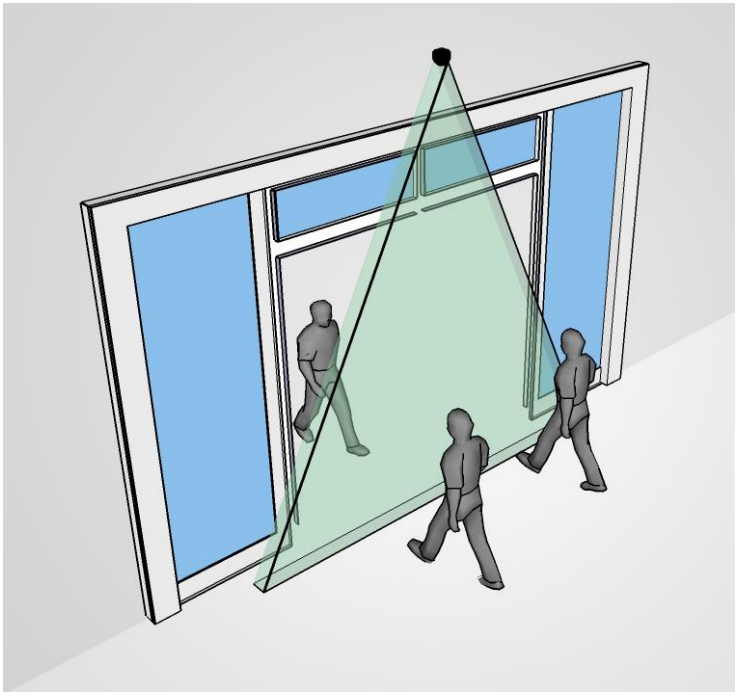
2.3. Brojač osoba zasnovan na Wi-Fi mreži

Brojanje osoba zasnovano na WiFi mreži koristi WiFi prijemnik za prikupljanje jedinstvenih WiFi upravljačkih okvira emitiranih od strane pametnih telefona s rasponom do 100 metara. Iako svi ljudi ne posjeduju pametni telefon, WiFi brojanje može proizvesti statistički značajne podatke o broju osoba u prostoriji zbog dostupnosti velikog uzorka.

Uobičajeno prilikom kretanja s uređajem s mogućnošću WiFi-a moguće je očitati MAC adresu uređaja. Ova adresa telefona, koja ima jedinstvenu i obično trajnu vrijednost, može biti upotrebljena u svrhu brojanja osoba u prostoriji. Prednost ovakvog pristupa brojanju osoba je u tome što osobe već posjeduju uređaje pomoću kojih su prebrojane. Ipak, noviji mobilni operacijski sustavi poput Apple iOS9 i Android 6.0 Marshmallow koriste agresivnije MAC rotacijske sheme i tako mijenjaju MAC adrese telefona, što čini brojače zasnovane na bežičnoj mreži nepraktičnim u budućnosti.

2.4. Brojači osoba zasnovani na kameri

Ovi brojači osoba sastoje se od kamere kao senzora spojene na ugradbenu platformu i programske podrške. Programska podrška za razvoj ovakvih brojača složenija je od one do sad navedenih i sastoji se od razvojnog okruženja, programskog jezika i biblioteka potrebnih za računalnu obradu slike. Postoje razni algoritmi za izvedbu ovakvih brojača, a neki od njih zasnivaju se na razlici između uzastopnih okvira, računanju optičkog toka slike ili strojnom učenju obrazaca koje računalo treba prepoznati. Nedostaci ovakvih brojača su u složenosti izvedbe programskog rješenja. Među prednosti svakako spadaju točnost brojanja i primjenjivost na unutarnje i vanjske lokacije.



Sl.2.3. *Brojač osoba zasnovan na kameri* [1]

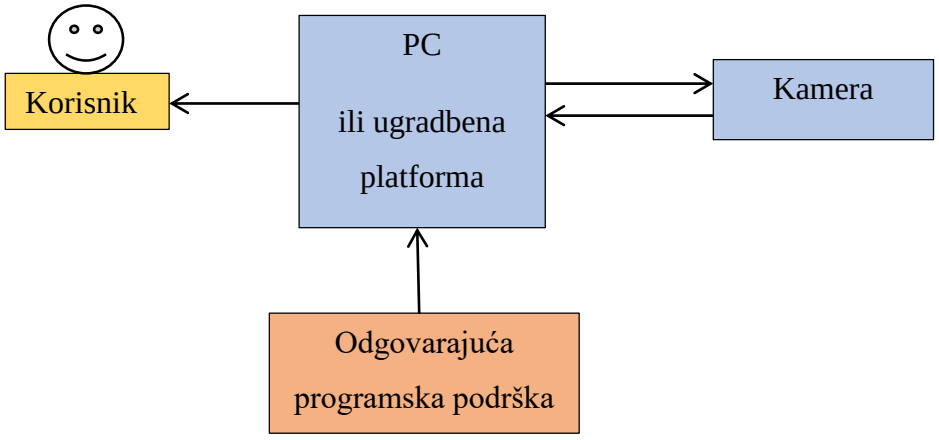
Na slici 2.3. prikazan je položaj kamere i regije interesa koju kamera snima u prostoru. Prolaskom kroz regiju interesa osobe budu detektirane, praćene i prebrojane ovisno o kretanju.

3. PRIJEDLOG BROJAČA OSOBA ZASNOVANOG NA KAMERI

Nakon proučavanja metoda brojanja osoba, a u sprezi s prednostima i nedostacima spomenutih metoda, za prijedlog rješenja odabran je brojač osoba zasnovan na kameri. Problematika opisanog rješenja može se podijeliti u dva funkcijska odjeljka:

1. Sklopovlje potrebno za razvoj i izradu brojača osoba zasnovanog na kameri
2. Programska podrška za praćenje i brojanje osoba u videu

Na slici 3.1. prikazan je blok dijagram rješenja brojača osoba. Kamera je senzor koji prikuplja slike iz vanjskog svijeta i šalje ih na obradu računalu koje može biti osobno računalo ili ugradbena platforma kao što je Raspberry Pi. Kamera i računalo su sklopovski dio prijedloga rješenja i zato su obojani plavom bojom na dijagramu. Slika se na računalu obrađuje pomoću odgovarajuće programske podrške (crveno). Nakon obrade moguće je krajnjem korisniku pružiti odgovarajuće informacije poput obrađene slike s označenim objektima ili broj osoba u svakom trenutku u prostoru koji kamera pokriva (žuto).



Sl.3.1. *Blok dijagram planiranog rješenja brojača osoba* [2]

3.1. Sklopovlje brojača osoba zasnovanog na kameri

Sklopovlje brojača osoba sastoji se od senzora za prikupljanje slika, tj. kamere i računalne jedinice na kojoj se obavlja daljnja obrada poslanih informacija. U prijedlogu rješenja odabrana je USB kamera zbog jednostavnosti povezivanja putem USB porta. Kao računalne jedinice odabrani su prijenosno računalo ili ugradbena platforma Raspberry Pi zbog pristupačnosti cijene i nabave i jednostavnosti rukovanja zbog dimenzija.

3.1.1. Senzor - kamera

Kao senzor za prikupljanje podataka iz stvarnog svijeta odabrana je USB WEB CAMERA CANYON, model **CNR-FWC120H.** Senzorska rezolucija kamere je 2 Mpixel, tehnologije senzora slike ¼ CMOS, maksimalne video rezolucije 1600x1200 i 30fps (640x480) okvira po sekundi.

Sl.3.2. *CANYON CNR-FWC120H*

3.1.2. Osobno računalo

Za računalo na kojemu se odvija programska obrada slike odabrano je prijenosno računalo ACER Aspire E1-510 sljedećih specifikacija:

CPU: Intel Pentium N3520 / 2.166 GHz

Max Turbo Speed: 2.42 GHz

Number of Cores: Quad-Core

Cache: L2 - 2 MB

RAM: 8GB DDR3 L

GPU: Intel HD Graphics

3.1.3. Raspberry Pi

Raspberry Pi je niskobudžetno računalo na jednoj pločici (eng. *single board computer)* veličine kreditne kartice. Iako je skromnijih mogućnosti od standardnih stolnih računala, na njemu je moguće obavljati niz operacija poput povezivanja s Internetom, komunikacija s raznim senzorima i programiranje računalne obrade slike. Zbog svoje male potrošnje, malih dimenzija, posjedovanja wifi adaptera i usb priključka pogodno je za razvoj ovakvih brojača osoba zasnovanih na kameri [8].

Iako postoji više inačica ovog uređaja za potrebe diplomskog rada korišten je Raspberry Pi 3 sljedećih specifikacija:

- SoC: Broadcom BCM2837.
- CPU: 4× ARM Cortex-A53, 1.2GHz.
- GPU: Broadcom VideoCore IV.
- RAM: 1GB LPDDR2 (900 MHz)
- Networking: 10/100 Ethernet, 2.4GHz **802.**11nwireless.

- Bluetooth: Bluetooth 4.1 Classic, Bluetooth Low Energy.
- Storage: microSD.

3.2. Programska podrška brojača osoba zasnovanog na kameri

Programska podrška brojača zasnovanog na kameri sastoji se u prvom redu od alata potrebnih za razvoj algoritama za obradu slike, a to su programski jezik, razvojno okruženje i posebne biblioteke korištene za programsku obradu slike. Osim alata programskoj podršci pripadaju i sami algoritmi za prepoznavanje i praćenje kretajućih objekata u video zapisu te algoritam osmišljen za brojanje osoba u video zapisu.

3.2.1. Programski alati za razvoj programske podrške

Programski alati i biblioteke korišteni za razvoj programske podrške prijedloga rješenja su:

1. OpenCV biblioteka za računalnu obradu slike
2. Microsoft Visual Studio razvojnog okruženje
3. C++ programski jezik

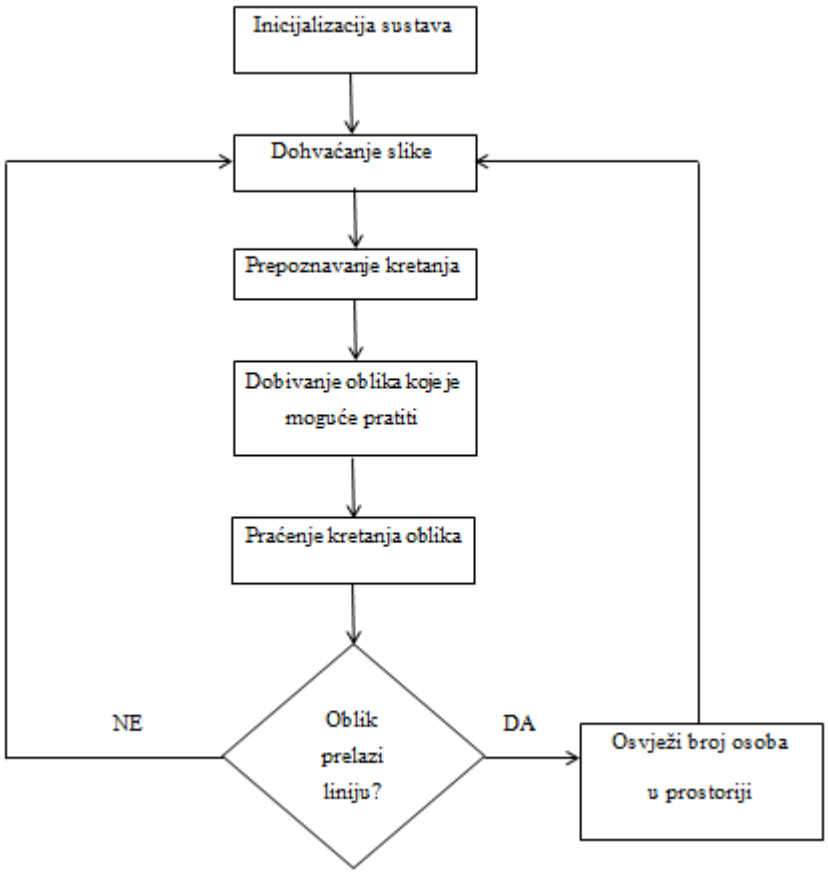
OpenCV (eng. *Open Source Computer Vision Library*) je *open source* biblioteka za računalni vid i strojno učenje. OpenCV je osmišljena da osigura zajedničku infrastrukturu za aplikacije računalnog vida i ubrza korištenje strojne percepcije u komercijalnim proizvodima. Budući da je BSD licencirani proizvod, OpenCV olakšava poduzećima koristiti i modificirati kôd [9]. Iako je ova biblioteka dostupna u mnogim izdanjima i verzijama, u ovom radu korištena je verzija OpenCV 3.2.0.

Microsoft Visual Studio je bogato, integrirano razvojno okruženje za stvaranje aplikacije za Windows, Android i iOS, kao i modernih web-aplikacija i usluga u oblaku [10]. Visual studio je IDE (eng. Integrated Development Enviroment) u kojemu je napisan kod rješenja ovog rada. Iako se ovo razvojno okružje pojavljuje u mnogim izdanjima i verzijama, u ovom radu korištena je verzija Microsoft Visual studio 2015 Community with Blend.

C++ je programski jezik opće namjene i srednje razine s podrškom za objektno orijentirano programiranje. Prvotno je razvijan u Bell Labs (laboratoriju telekomunikacijske tvrtke Bell) pod rukovodstvom Bjarnea Stroustrupa tokom 1980-ih, i to kao proširenje programskom jeziku C pa mu je originalno ime bilo "C with classes" (eng. C s klasama). Zbog velike potražnje za objektno orijentiranim jezicima te izrazitim sposobnostima istih, specifikacija programskog jezika C++ ratificirana je 1998. kao standard ISO/IEC 14882 [11]. C++ je programski jezik u kojem je izrađena cjelokupna programska podrška brojača osoba zasnovanog na kameri.

3.2.2. Prepoznavanje i praćenje kretajućih objekata u videu

Nakon inicijalizacije sustava i hvatanja okvira poslanog kamerom započinje programska obrada slike. U ovom potpoglavlju bit će opisane najvažnije funkcije biblioteka OpenCV koje su korištene u algoritmu prepoznavanja i praćenja kretajućih objekata u video zapisu.



Sl.3.3. *Blok dijagram programskog rješenja brojača osoba*

Prepoznavanje kretanja i praćenje kretajućih oblika obavlja se u trećem, četvrtom i petom koraku dijagrama. Dijagram počinje s inicijalizacijom sustava i dohvaćanjem slike iz kamere. Nakon što je kretanje uspješno izolirano pomoću funkcije apsolutne razlike između okvira ili funkcije izračunavanja optičkog toka u trećem koraku, daljnjom obradom slike dobiven je oblik koji možemo izmjeriti i pratiti u četvrtom. U petom je nastavljeno praćenje oblika uspoređujući ih s već poznatim oblicima iz prethodnih okvira da bi se u šestom koraku izvršila provjera prelaska zamišljene linije koja razdvaja prostor.

Prepoznavanje kretanja

Prepoznavanje kretanja izvedeno je pomoću funkcija cv::absdiff i cv::calOpticalFlowFarneback. U ovom dijelu potpoglavlja bit će navedeni primjeri i objašnjenja korištenja ove dvije funkcije u tu svrhu. Primjer korištenja cv::absdiff funkcije u kodu rješenja:

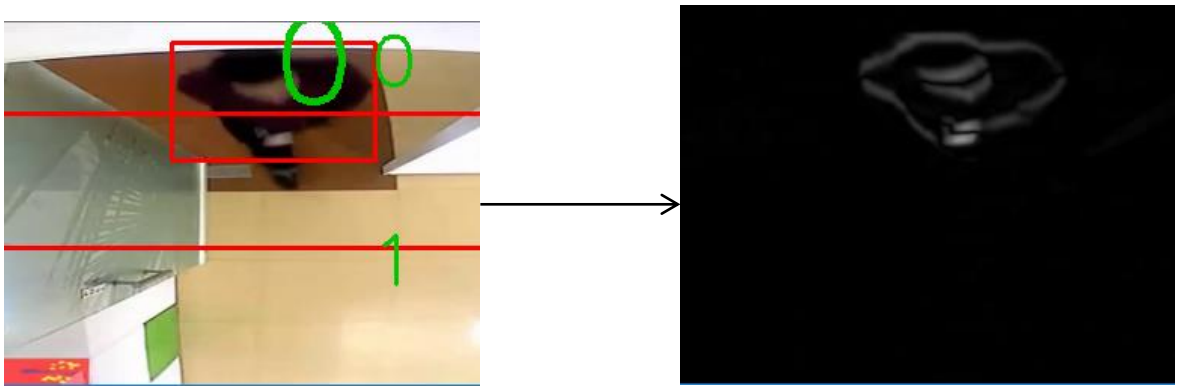
```
cv::absdiff(imgFrame1Copy, imgFrame2Copy, imgDifference);
```

Izračunava po elementu apsolutnu razliku između dva polja ili između polja i skalara. Funkcija cv::absdiff izračunava apsolutnu razliku između dva polja kad imaju istu veličinu i tip :

$$dst(I)=saturate(|src1(I)-src2(I)|) \quad (3\text{-}1)$$

gdje je *dst(I)* izlazno polje (dvodimenzionalni vektor), *src1(I)* prvo ulazno polje, a *src2(I)* drugo ulazno polje funkcije, a *saturate* je funkcija popunjavanja izlaznog polja vrijednostima apsolutne razlike dvaju ulaznih polja. Ulazna polja vrijednosti su okviri (eng. *frame)* koje kamera šalje računalu.

Ova funkcija korištena je u svrhu prepoznavanja kretanja između dva okvira u trećem koraku iz dijagrama prijedloga rješenja. Budući da su dva uzastopna okvira koja kamera pošalje računalu dva polja vrijednosti, ova funkcija se nameće kao svrsishodna. Razlika između okvira prije pojave kretanja i okvira u kojemu se kretanje pojavilo najčešće je objekt koji se kreće. Dakle, sve promjene koje se nalaze u dugom okviru, a nisu bile prisutne u prvom, smatraju se kretanjem i daljnjim provjerama dimenzija i omjera dimenzija objekta koji se kreće vrši se klasifikacija na osobe i druge objekte.



a) b)

Sl. 3.4. *a) Analizirani oblik,* b) *rezultat cv::absdiff između analiziranog okvira i prethodnog*

Drugi princip prepoznavanja kretanja zasniva se na optičkom toku. Izračunavanje optičkog toka za svaki piksel po algoritmu Gunnara Farnebacka u predloženom rješenju radi se pomoću sljedeće funkcije:

```
void cvCalcOpticalFlowFarneback(const CvArr* prevImg,
                                const CvArr* nextImg,
                                CvArr* flow,
                                double pyrScale,
                                int levels,
                                int winsize,
                                int iterations,
                                int polyN,
                                double polySigma,
                                int flags)
```

Ova funkcija također je korištena u svrhu prepoznavanja kretanja između dva okvira u trećem koraku iz dijagrama prijedloga rješenja. Za svaki piskel prethodnog okvira funkcija pronalazi novu lokaciju u trenutnom okviru i na temelju tih podataka daje intenzitet i smjer kretanja piksela.

Funkcija prima 10 parametara:

1) `prevImg` – prva 8-bitna jednokanalna ulazna slika
2) `nextImg` – druga ulazna slika istih karakteristika kao i prva
3) `flow` – izlazna slika rezultata tipa CV_32FC2

4) `pyrScale` - Parametar koji određuje skalu slike (<1) za izradu piramida za svaku sliku. `pyrScale` = 0.5 znači klasičnu piramidu, gdje je svaki sljedeći sloj dvaput manji od prethodnog. U kodu rješenja korištena je vrijednost 0,4.
5) `levels` - Broj slojeva piramide, uključujući početnu sliku. Levels = 1 znači da se ne izrađuju dodatni slojevi i koriste se samo izvorne slike. U kodu rješenja korištena je vrijednost 1.
6) `winsize` - Prosječna veličina prozora. Veće vrijednosti povećavaju robusnost algoritma na šum slike i daju veće šanse za otkrivanje brzih pokreta, ali daju više mutnih oblika pokreta. U kodu rješenja korištena je vrijednost 8.
7) `iterations` - Broj iteracija koje algoritam radi na svakoj razini piramide. U kodu rješenja korištena je vrijednost 2.
8) `polyN` - Veličina pikselskog susjedstva koja se koristi za pronalaženje polinoma ekspanzije u svakom pikselu. Veće vrijednosti znače da će slika biti aproksimirana s glatkim površinama, što daje robusniji algoritam i više mutnih oblika gibanja. U kodu rješenja korištena je vrijednost 1.
9) `polySigma` - Standardna devijacija prilikom proširenje polinoma. U kodu rješenja korištena je vrijednost 0,2.
10) `flags` – zastavice. U kodu rješenja korištena je vrijednost 0.

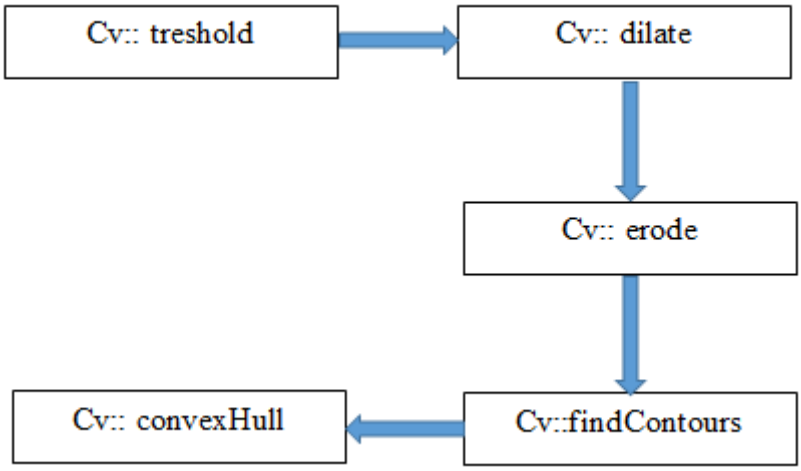
Funkcija pronalazi optički protok za svaki piksel *prevImg* slike pomoću Farneback algoritma tako da vrijedi:

$$prevImg(y,x) \sim nextImg\,(y + flow(y,x)[1],\ x + flow(y,x)[0]) \qquad (3\text{-}2)$$

Svaki piksel prethodnog okvira tj. *prevImg* slike opisan je zbrojem položaja na sljedećem okviru tj. *nextImg* i optičkog toka (*flow).*

Dobivanje oblika koje je moguće pratiti

Dobivanje oblika koje je moguće pratiti izvedeno je pomoću funkcija cv::treshold, cv::dilate, cv::erode, cv::findContours i cv::convexHull. U ovom dijelu potpoglavlja bit će navedeni primjeri i objašnjenja korištenja ovih funkcija u tu svrhu.



Sl.3.5. *Dijagram s funkcijama korištenim za dobivanje oblika koje je moguće pratiti*

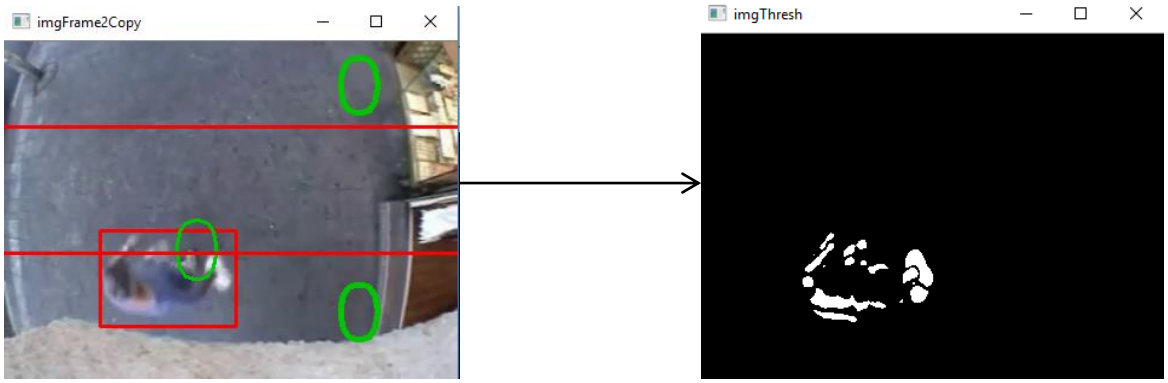
Slika 3.5. prikazuje redoslijed obavljanja funkcija prilikom dobivanja oblika koje je moguće pratiti. Nakon što je kretanje u okviru primijećeno ono se izolira od statičnog dijela okvira funkcijom cv::treshold. Zatim se popunjavaju praznine između kretajućih dijelova funkcijom cv::dilate i smanjuje ukupna površina funkcijom cv::erode. Na kraju se određuju granice kretajućeg oblika funkcijom cv::findContours i zaokružuje u jednu cjelinu funkcijom cv::convexHull.

Izdvajanje dijelova slike koji se kreću od ostatka kako bi se započelo dobivanje oblika koje je moguće pratiti obavljeno je funkcijom cv::treshold:

```
cv::threshold(inputArray src, outputArray dst, double tresholdValue, double maxValue,
                                    tresholdType);
```

Ako je vrijednost piksela veća od granične vrijednosti, dodjeljuje mu se jednu vrijednost (npr. bijelo), inače se dodjeljuje drugu vrijednost (npr. crno). Prvi argument je *source* slika na kojoj se primjenjuje funkcija. Drugi argument je slika odredište u koju će biti spremljen rezultat. Treći argument je granična vrijednost koja se koristi za razvrstavanje piksela, u rješenju 30. Četvrti argument je *maxVal* koja predstavlja vrijednost koja se daje ako vrijednost piksela iznosi više od granične vrijednosti, u rješenju 255 (vrijednost bijele boje). OpenCV nudi različite stilove thresholdinga, a to je odlučeno zadnjim parametrom funkcije. U rješenju je korišten tip *CV_THRESH_BINARY* koji dodjeljuje vrijednost 0 pikselima s vrijednošću manjom od

tresholdValue, a *maxValue* pikselima s vrijednošću jednakom ili većom od *tresholdValue*. Ova funkcija korištena je u četvrtom koraku iz dijagrama prijedloga rješenja [5].



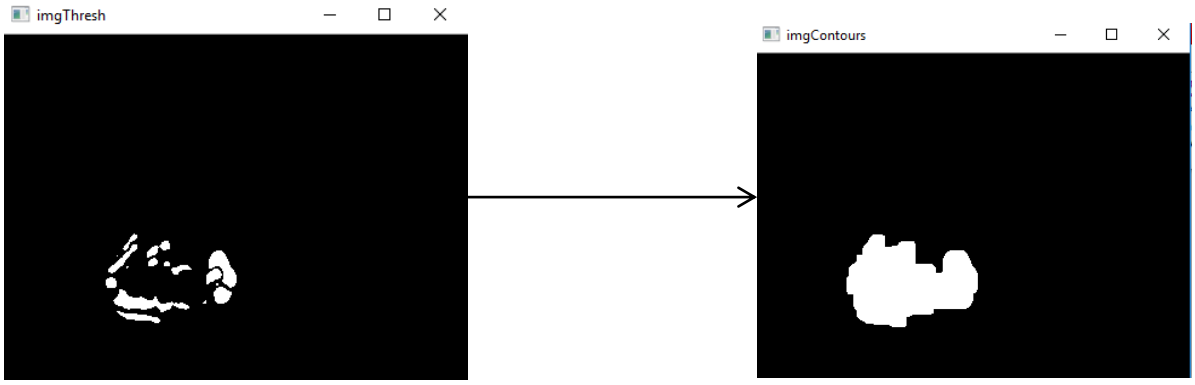
Sl.3.6. *Rezultat cv::treshold funkcije*

Nakon izdvajanja dijelova slike koji se kreću potrebno je naći granicu oblika koji će biti praćen, tj. krivulju koja okružuje te dijelove. To je obavljeno funkcijom cv::findContours:

```
void cv::findContours ( InputOutputArray     image,
                        OutputArrayOfArrays  contours,
                        int                  mode,
                        int                  method,)
```

Prvi argument je ulazna slika, drugi je matrica u koju je spremljen rezultat, treći jedan od načina rada funkcije (u rješenju vrijednost ovog argumenta je *cv::RETR_EXTERNAL*) i četvrti je metoda aproksimacije za nalaženje kontura (vrijednost iz rješenja je *cv::CHAIN_APPROX_SIMPLE).*

Konture se može jednostavno objasniti kao krivulju koja spaja sve stalne točke (uz granicu), ima istu boju i intenzitet. Konture su koristan alat za analizu oblika i prepoznavanje objekta i granica. Ova funkcija korištena je u četvrtom koraku iz dijagrama prijedloga rješenja.

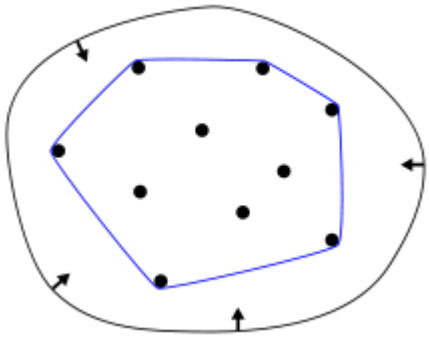


Sl.3.7. *Rezultat cv::findContours funkcije*

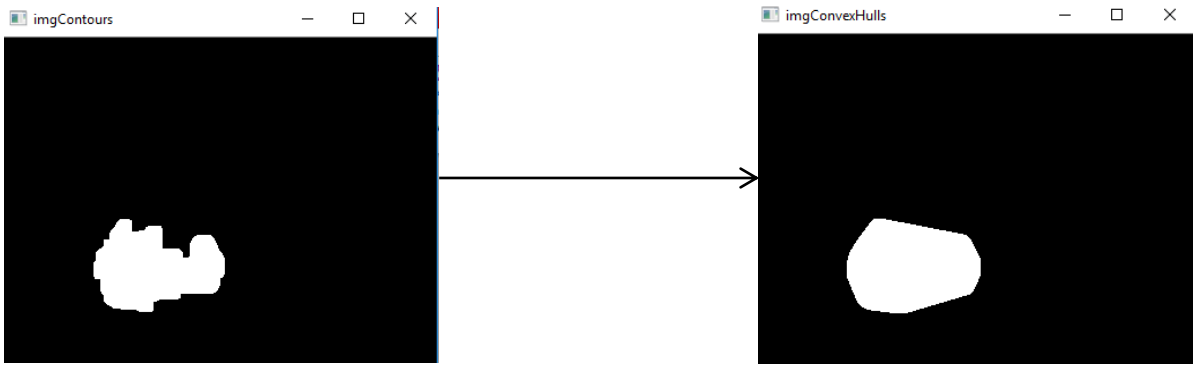
Nakon dobivanja kontura praćenog oblika potrebno je sve njegove točke spojiti u jednu cjelinu. To je obavljeno funkcijom cv::convexHull. Primjer korištenja ove funkcije u kodu rješenja:

```
for (unsigned int i = 0; i < contours.size(); i++) {
                cv::convexHull(contours[i], convexHulls[i]);
        }
```

Konveksna ljuska polja *X* točaka u euklidskoj ravnini je najmanji konveksan skup koji sadrži *X*. Formalno, konveksna ljuska se može definirati kao sjecište svih konveksnih skupova koji sadrže *X* ili kao skup svih konveksnih kombinacija točaka u *X* [6]. Ova funkcija korištena je u svrhu dobivanja zatvorenih jednostavnih oblika u četvrtom koraku iz dijagrama prijedloga rješenja.

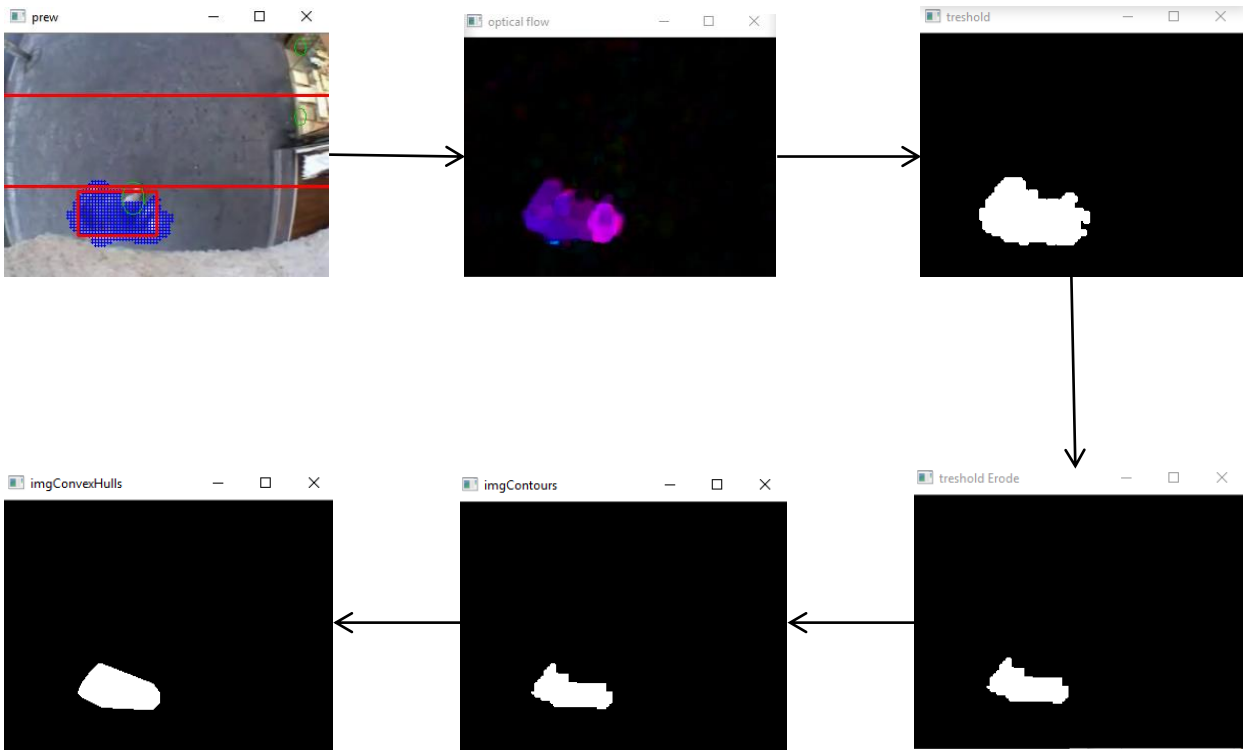


Sl.3.8. *Konveksna ljuska skupa točaka X (obrubljeno plavom bojom)* [7]



Sl.3.9. *Rezultat cv::findContours funkcije*

Čitav proces dobivanja oblika koje je moguće pratiti prikazan je slikom 3.10.



Sl.3.10. *Faze obrade slike kod prepoznavanja i praćenja kretajućih oblika*

3.2.3. Brojanje osoba

Budući da je svaki okvir koji kamera šalje računalna matrica piksela s točno određenom pozicijom položaja piksela u obliku (*X*,*Y*), prateći koordinatu *Y* središta objekta kojeg pratimo iz okvira u okvir moguće je odrediti kada objekt prelazi određenu granicu kao na primjeru danom na slici 3.6. lijevo. Koordinata *X* označava stupce matrice, a koordinata *Y* redove, što znači da uspoređujući *Y* vrijednost centra objekta kojeg pratimo s graničnom vrijednosti možemo pratiti njegovo vertikalno kretanje, tj. kada objekt prelazi horizontalnu zamišljenu liniju. Drugim riječima ako je u dva uzastopna okvira poslana kamerom u jednom *Y* vrijednost centra praćenog objekta manja od granične vrijednosti, a u drugom veća ili jednaka, objekt je vertikalnim kretanjem prešao zadanu granicu.

Kod opisane funkcije iz rješenja:

```
Bool	blnAtLeastOneBlobCrossedTheLine	=	checkIfBlobsCrossedTheLine(blobs,
intHorizontalLinePosition,intHorizontalLinePosition2, count);

bool checkIfBlobsCrossedTheLine(std::vector<Blob> &blobs, int &intHorizontalLinePosition,
int &intHorizontalLinePosition2, int &Count) {
	bool blnAtLeastOneBlobCrossedTheLine = false;

	for (auto &blob : blobs) {

		if (blob.blnStillBeingTracked == true && blob.centerPositions.size() >= 2)
{
			int prevFrameIndex = (int)blob.centerPositions.size() - 2;
			int currFrameIndex = (int)blob.centerPositions.size() - 1;

			if (blob.centerPositions[prevFrameIndex].y >
intHorizontalLinePosition && blob.centerPositions[currFrameIndex].y <=
intHorizontalLinePosition) {
				Count++;
				blnAtLeastOneBlobCrossedTheLine = true;
			}
			if (blob.centerPositions[prevFrameIndex].y <
intHorizontalLinePosition2 && blob.centerPositions[currFrameIndex].y >=
intHorizontalLinePosition2) {
				Count--;
				blnAtLeastOneBlobCrossedTheLine = true;
			}
		}

	}

	return blnAtLeastOneBlobCrossedTheLine;
}
```

Funkcija prima 4 argumenta: popis objekata koji se prate iz okvira u okvir, *X* koordinata linije koja broji ulaske, *X* koordinata linije koja broji izlaske i trenutni broj izbrojanih osoba.

Najprije se iz popisa praćenih objekata izdvajaju oni koji se još uvijek prate i poznat im je položaj centra:

```
if (blob.blnStillBeingTracked == true && blob.centerPositions.size() >= 2)
```

Nakon izdvajanja željenih objekata određuju se indeksi zadnjeg i predzadnjeg okvira:

```
int prevFrameIndex = (int)blob.centerPositions.size() - 2;
int currFrameIndex = (int)blob.centerPositions.size() - 1;
```

Potom provjera je li objekt prošao liniju koja broji ulaske u prostor od interesa usporedbom *Y* koordinate njegova centra u dva uzastopna okvira s vrijednost *Y* koordinate točaka na liniji:

```
if (blob.centerPositions[prevFrameIndex].y > intHorizontalLinePosition &&
blob.centerPositions[currFrameIndex].y <= intHorizontalLinePosition) {
                        Count++;
                        blnAtLeastOneBlobCrossedTheLine = true;
                }
```

Potom provjera je li objekt prošao liniju koja broji izlaske iz prostorije usporedbom *Y* koordinate njegova centra u dva uzastopna okvira s vrijednost *Y* koordinate točaka na liniji:

```
if (blob.centerPositions[prevFrameIndex].y < intHorizontalLinePosition2 &&
blob.centerPositions[currFrameIndex].y >= intHorizontalLinePosition2) {
                        Count--;
                        blnAtLeastOneBlobCrossedTheLine = true;
                }
```

Naposljetku vraćanje bool vrijednosti koja označava prolazak preko jedne od linija brojanja s 1.

```
return blnAtLeastOneBlobCrossedTheLine;
```

4. REZULTATI MJERENJA

Mjerenje je obavljeno pomoću dva različita programa i pet različitih videa [12 - 16]. U srži prvog programa je funkcija 4.3.1. cv:: absdiff i funkcija brojanja, dok se drugi temelji na funkciji 4.3.2. cv::calcOpticalFlowFarneback i funkciji brojanja. Ostale funkcije koriste se u oba programa s prilagođenim parametrima za svaku od kombinacija. Eksperimenti su izvedeni pomoću osobnog računala i kamere kako je navedeno u potpoglavljima 3.1. i 3.2. Točnost prebrojavanja određena je kvocijentom izlazne vrijednosti rješenja i ručnog prebrojavanja i izražena u postocima. Testni video zapisi imaju rezoluciju 480 x 480 piksela. Video zapisi imaju različite karakteristike poput broja osoba koje se istovremeno kreću razmatranim područjem, različit kut snimanja kamere što u potpunosti mijenja geometriju okvira i sl. U tablici 4.1. navedeni su video zapisi i njihove karakteristike, kao i rezultati mjerenja. U prvom retku navedena je karakteristika položaj kamere koji može biti vertikalan i horizontalan. O ovoj karakteristici ovisi čitava geometrija okvira što utječe na funkcije prepoznavanja kretanja kao i na provedbu samog brojanja. U drugom retku je kratki opis onoga što se prikazuje u video zapisu u smislu karakteristika koje će biti testirane. Nakon toga u iduća dva retka navedene su točnosti prebrojavanja po videu za obje funkcije prepoznavanja kretanja u postocima. U sljedećem retku je karakteristika broj okvira po sekundi (eng. *frame per second*) za svaki video. U zadnja dva retka tablice su brzine izvođenja algoritama za pojedini okvir izražene u milisekundama.

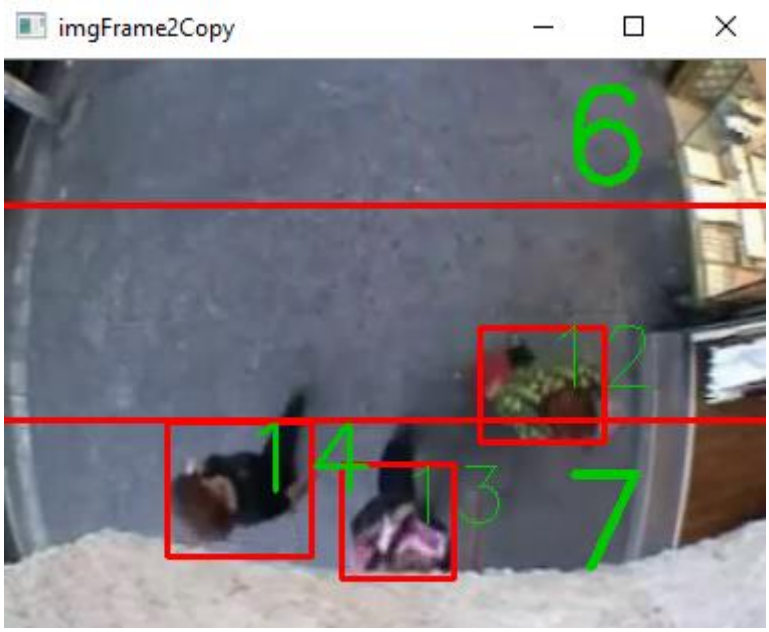
	Video 1	Video 2	Video 3	Video 4a	Video 4b	Video 5
Položaj kamere	vertikalan	vertikalan	Vertikalan	horizontalan - bočni	horizontalan- frontalni	horizontalan - bočni
Sadržaj	Široki prolaz na ulici s visokom frekvencijom pješačkog, biciklističkog i motociklističkog prometa	Uski prolaz (vrata) u zatvorenom prostoru s visokom frekvencijom prolazaka i promjenama razine osvjetljenja	Uski prolaz na otvorenom s niskom frekvencijom pješačkog prometa	Široki prolaz na ulici s visokom frekvencijom pješačkog prometa i slabo fiksiranom kamerom	Široki prolaz na ulici s visokom frekvencijom pješačkog prometa	Široki prolaz na uglu ulice s niskom frekvencijom pješačkog
Točnost cv::absdiff	96%	100%	100%	58,3%	-	100%
Točnost optičkog toka	96%	100%	100%	-	50%	75%
fps	25	25	30	30	24	25
Primjer lokacije	Pločnik ispred trgovačkog centra	Ured s velikim brojem zaposlenih	Ulazna vrata kuće	Ulica , Park	Ulica , Park	Ugao na ulici
Vrijeme obrade jednog okvira sa cv::absdiff	45ms	30ms	34ms	29ms	-	17ms
Vrijeme obrade jednog okvira s optičkim tokom	74ms	76ms	78ms	-	84ms	73ms

Tablica 4.1. *Rezultati mjerenja*

4.1. Rezultati programa zasnovanog na funkciji cv::absdiff

4.1.1. Test video 1

Cilj: Ispitati točnost prebrojavanja na vanjskim mjestima širokog prostora prolaska, visoke frekvencije i niske razine promjene osvjetljenja. Ispitati točnost prebrojavanja biciklističkog i motociklističkog prometa uz pješački.



Sl.4.1. *cv::absdiff i video 1 rezultat*

Na slici 4.1. prikazan je okvir iz videa 1 na kojim se vide granične linije brojanja za oba smjera. Objekti koji se prate omeđeni su crvenim pravokutnikom i označeni id brojem. Trenutni rezultati brojanja za svaku liniju napisani su pored nje. Postavljanjem kuta kamere vertikalno u odnosu na površinu prolaska objekata praćenja te uz niske razine promjene osvjetljenja program zasnovan na funkciji cv::absdiff ostvaruje 96% točnosti prebrojavanja. Osim pješačkog i ostali promet na videu uspješno je prebrojan uz male preinake u vrijednostima očekivane veličine objekata praćenja. U širokim prostorima najveći izazov za prebrojavanje dolazi u slučajevima uskog mimoilaženja objekata kako za ovu verziju algoritma rješenja tako i za onu zasnovanu na

optičkom toku. Ovaj izazov je riješen testiranjem vrijednosti s kojom smanjujemo ukupnu površinu praćenog objekta putem funkcije cv:: erode.

4.1.2. Test video 2

Cilj: Ispitati točnost prebrojavanja u unutarnjim mjestima uskog prostora prolaska, visoke frekvencije i visoke razine promjene svjetlosti.

Sl.4.2. *cv::absdiff i video 2 rezultat*

U primjerima situacija s uskim prostorom prolaska kao na slici 4.2. rjeđe dolazi do mimoilaženja osoba koje prolaze ispod brojača pa je zadatak za rješenje tim lakši. Specifičnost ovog videa je u osvjetljenju unutarnjeg prostora koje stvara sjenu prilikom prolaska. Budući da je program zasnovan na funkciji cv::absdiff puno manje „osjetljiv“ na promjene nižeg inteziteta (sjena) između dva okvira od onoga zasnovanog na optičkom toku to za njega ne predstavlja osobitu prepreku.

4.1.3. Test video 3

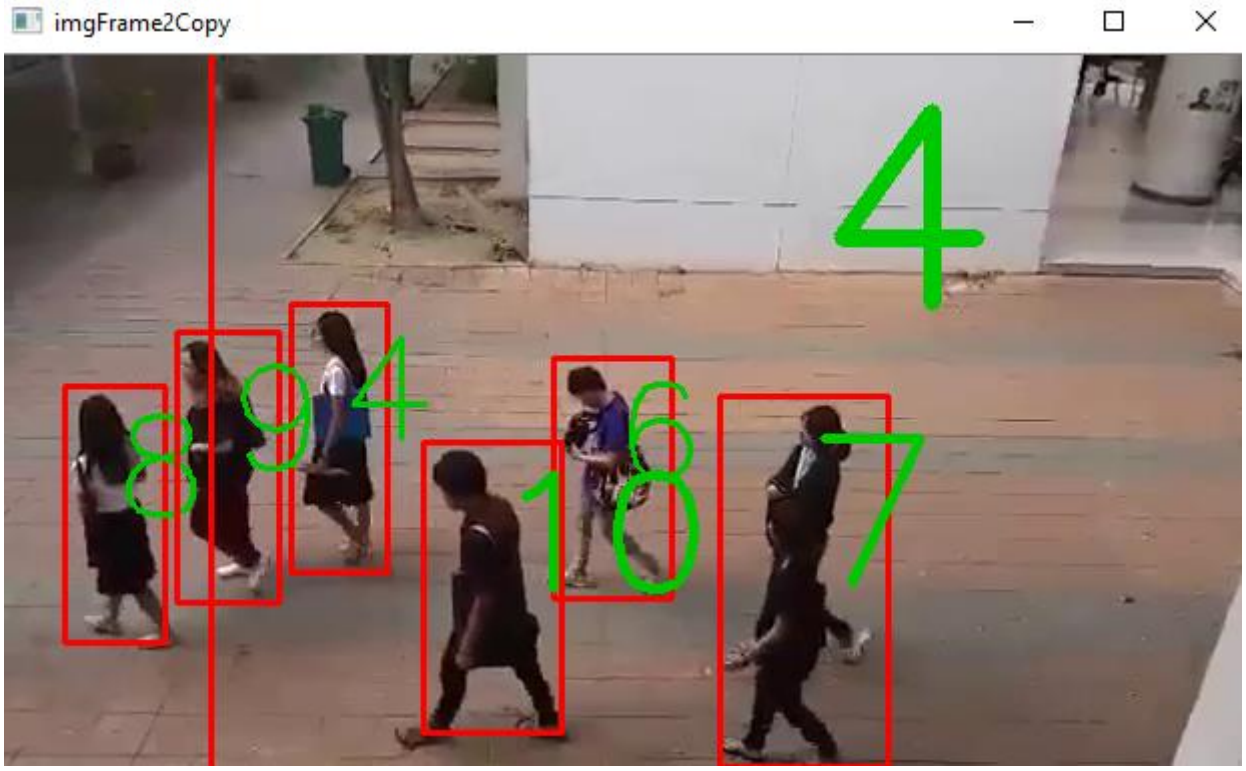
Cilj: Ispitati upotrebljivost rješenja za kućnu uporabu.

Sl.4.3. *cv::absdiff i treći video rezultat*

U primjerima situacija uskog prolaza i niske frekvencije prolazaka rješenje zasnovano na funkciji cv::absdiff ne nailazi na probleme pri prebrojavanju kao ni rješenje zasnovano na cv::calOpticalFlowFarneback. Obje metode postižu 100% učinkovitost, tj. ispravno su izbrojale 8 ulazaka odnosno izlazaka iz kuće.

4.1.4. Test video 4a

Cilj: Ispitati točnost prebrojavanja kada se promijeni kut snimanja kamere.



Sl.4.4. *cv::absdiff i video 4a rezultat*

Problemi kod ovog kuta montaže kamere su u tome što se objekti praćenja često nalaze jedan iza drugoga. Ovako preklapanje smanjuje vidljivost objekta i mogućnost razaznavanja točnih granica između dva objekta koja pratimo. Na slici 4.4. ovaj problem vidimo kod objekta s ID oznakom 7, dok su ostali objekti uspješno izolirani u tom kadru. U slijedećih nekoliko slika objašnjeni su razlozi nastanka ovog problema kod metode prepoznavanja kretanja zasnovane na funkciji cv::absdiff.

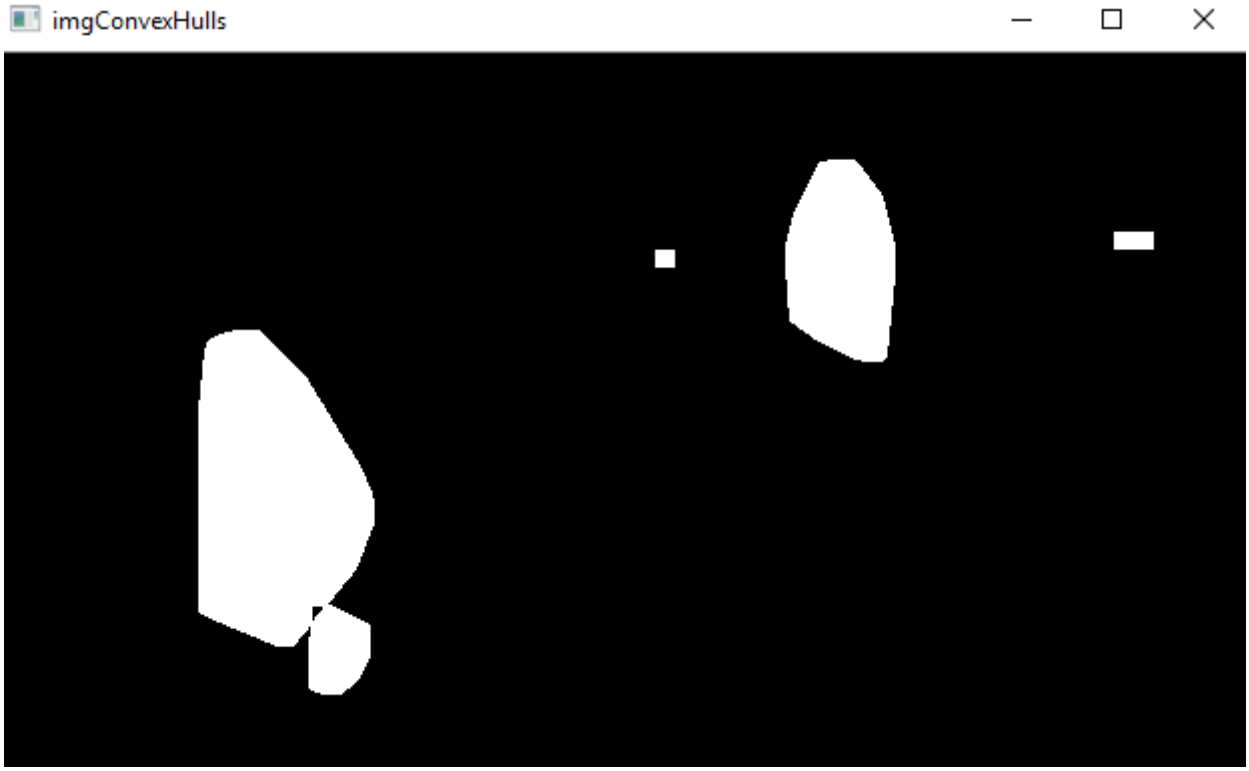
Sl.4.5. *Preklapanje objekata praćenja kod nevertikalnog kuta kamere*

Na slici 4.5. vide se četiri osobe u kadru i samo dva objekta praćenja. Razlog za to nalazi se u nemogućnosti fukcije cv::absdiff da jasno razazna koje razlike između dva uzastopna okvira pridaju prvoj a koje drugoj osobi.

Sl.4.6. *cv::absdiff slike 10*

Ovakav rezultat funkcije cv::absdiff izaziva lančanu reakciju u daljnim fazama obrade što je prikazano na slici 4.6. kao i na idućim slikama.

Sl.4.7. *Konture slike 10*



Sl.4.8. *cv::convexHull slike 10*

Budući da u preklapanju oblici koje pratimo u sebi često sadrže više osoba iz kadra kao na slici 4.8., točnost prebrojavanja rješenja obje verzije algoritma na ovom videu znatno je manja od prethodnih.

4.1.5. Test video 5

Cilj: Ispitati točnost prebrojavanja na vanjskim mjestima širokog prostora prolaska, niske frekvencije i niske razine promjene svjetlosti pri promijeni kuta kamere.

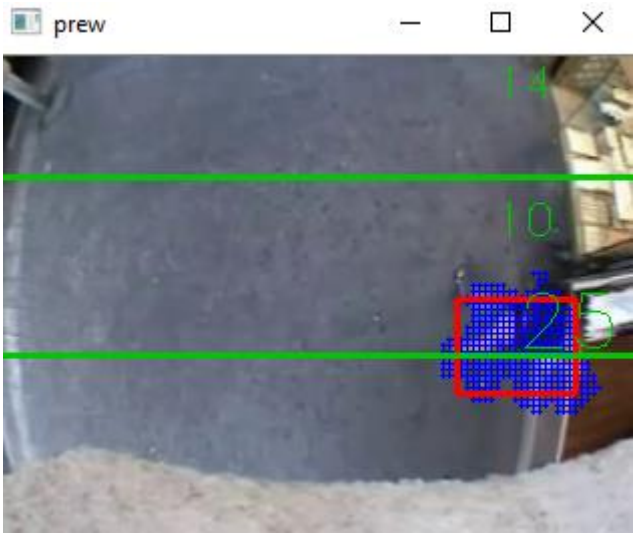
Sl.4.9. *cv::absdiff i video 5 rezultat*

Budući da je pokazano kako funkcija cv::absdiff ima velikih problema u preklapanju objekata praćenja, program je testiran na ovakvom jednostavnom videu bez mimoilaženja osoba u kadru pri čemu je kut kamere sličan kao kod test videa 4 (nevertikalan kut kamere). U ovakvim uvjetima točnost prebrojavanja se vraća na 100%.

4.2. Rezultati programa zasnovanog na funkciji cv::calcOpticalFlowFarneback

4.2.1. Test video 1

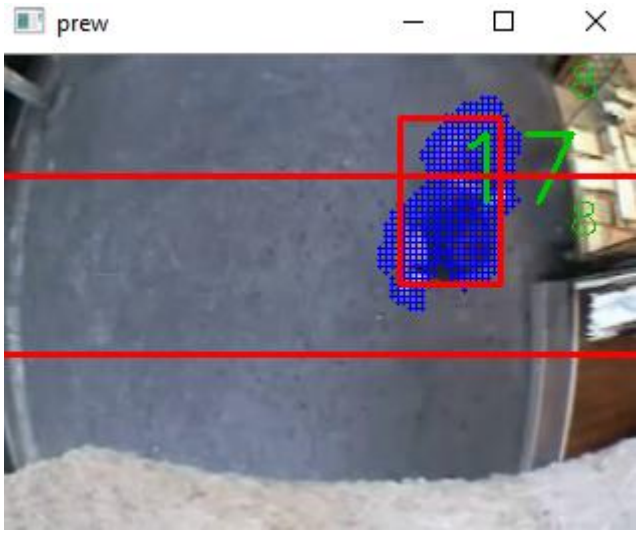
Cilj: Ispitati točnost prebrojavanja na vanjskim mjestima širokog prostora prolaska, visoke frekvencije i niske razine promjene svjetlosti. Ispitati točnost prebrojavanja biciklističkog i motociklističkog prometa uz pješački.



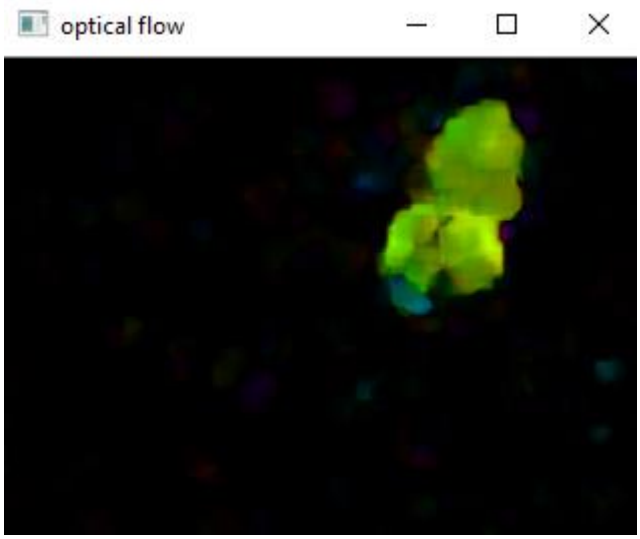
Sl.4.10. *cv::calcFlowFarneback i video 1 rezultat*

Na slici 4.10. vidi se trenutak brojanja oblika s ID oznakom 25 kako centralnim pikselom crvenog okružujućeg pravokutnika prelazi liniju brojanja za smjer prema gore. Plava rešetka koja prekriva lik osobe koja se prati je dio rezultata funkcije optičkog toka koji odgovara ciljanim uvjetima.

Problem koji se pojavio u prebrojavanju ovog videa je, kao i kod algoritma zasnovanog na apsolutnoj razlici, kada dvije osobe prelaze čitavom dužinom okvira držeći se jedno za drugo i istom brzinom pritom ostavljajući trag optičkog toka kao jedna cjelina kao što prikazuje slika 4.11.



Sl.4.11. *Dvije osobe prebrojane kao jedna*



Sl.4.12. *Optički tok slike 4.11.*

Na slici 4.12. vidi se rezultat funkcije optičkog toka koji ne vidi jasnu granicu između dvije osobe budući da su na slici kamere fizički spojene, kreću se gotovo istom brzinom i u istom smjeru. To pokreće lančanu reakciju nakon određivanja *treshold* slike iz koje se izrađuje objekt za praćenje.

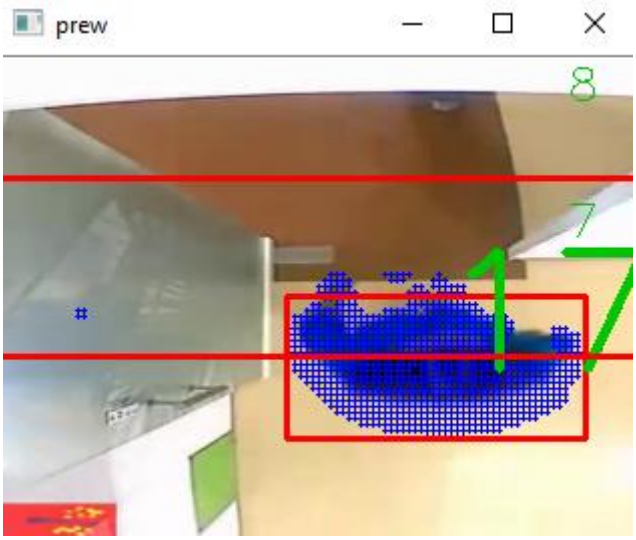
Sl.4.13. *cv::treshold slike 4.11.*

Sl.4.14. *Jedan objekt praćenja za dvije osobe*

Na slici 4.14. prikazan je jedan objekt praćenja za dvije osobe nastao lančanom reakcijom nakon primjene funkcije optičkog toka.

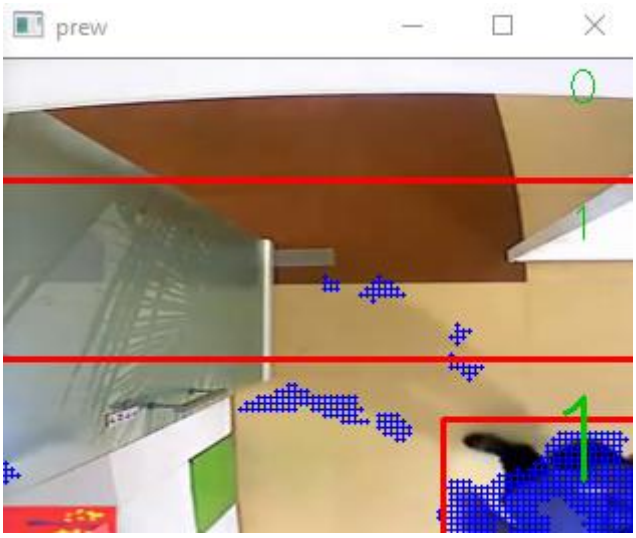
4.2.2. Test video 2

Cilj: Ispitati točnost prebrojavanja u unutarnjim mjestima uskog prostora prolaska, visoke frekvencije i visoke razine promjene svjetlosti.



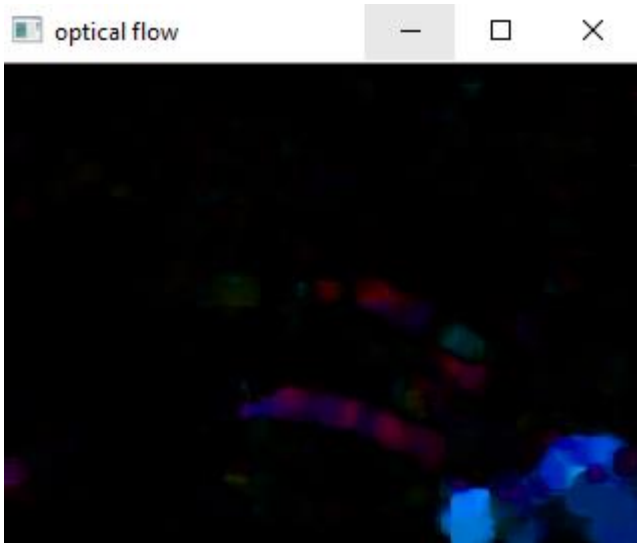
Sl.4.15. *cv:: calcOpticalFlowFarneback i video 2 rezulta*

Specifičnost ovog videa je u osvjetljenju prostorije koje izaziva izraženu pojavu sjene u okvirima. Budući da cv::calcOpticalFlowFarneback očitava kretanje piksela između dva okvira sjena predstavlja svojevrsni šum kojeg je potrebno minimizirati. To je učinjeno postavljanjem *treshold* vrijednosti na izlaznu vrijednost funkcije optičkog toka koja predstavlja magnitudu. Na ovaj način dijelovi slike s manjom magnitudnom vrijednosti optičkog toka (sjena) ostaju nedostupni u daljnjoj obradi. U kombinaciji s erodiranjem dijelova koji prolaze u daljnju obradu postavljanje *treshold* vrijednosti rješava šum koji predstavlja sjena.



Sl.4.16. *Minimizacija utjecaja sjene na objekt praćenja*

Na slici 4.16. vidljiva je prisutnost sjene u okviru kao i njeno očitanje funkcijom optičkog toka (plavi pikseli i slika 4.17.). Ipak, površina pokrivena sjenom nije uključena unutar granica oblika koji je praćen.

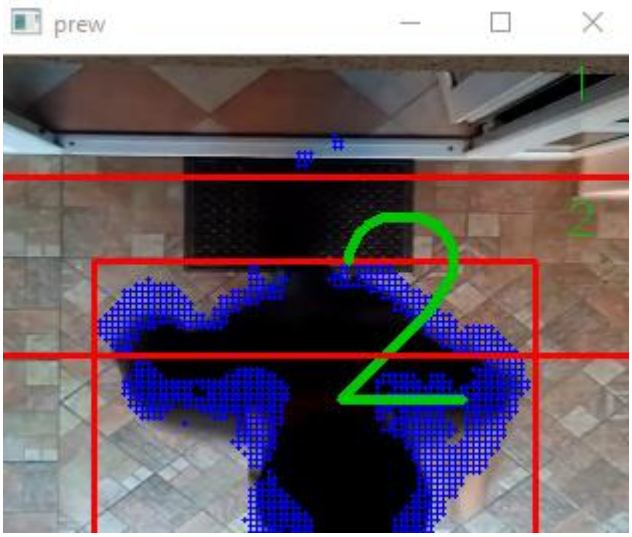


Sl.4.17. *cv::calcOpticalFlowFarneback slike 4.16.*

Prigušenije boje na površini pokrivenoj sjenom na izlazu funkcije optičkog toka ukazuju na slabiju jačinu optičkog toka u tom djelu okvira.

4.2.3. Test video 3

Cilj: Ispitati upotrebljivost rješenja za kućnu uporabu.



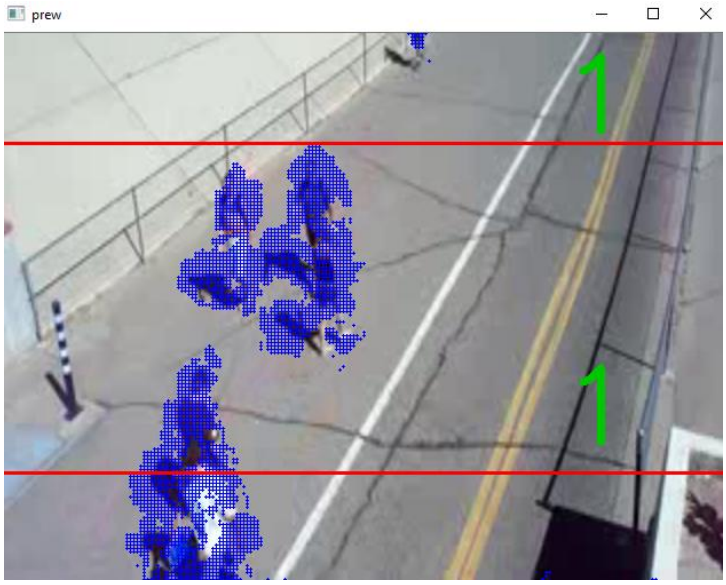
Sl.4.18. *cv:: calcOpticalFlowFarneback i video 3 rezultat*

U primjerima situacija uskog prolaza i niske frekvencije prolazaka rješenje zasnovano na funkciji optičkog toka ne nailazi na probleme pri prebrojavanju.

4.2.4. Test video 4b

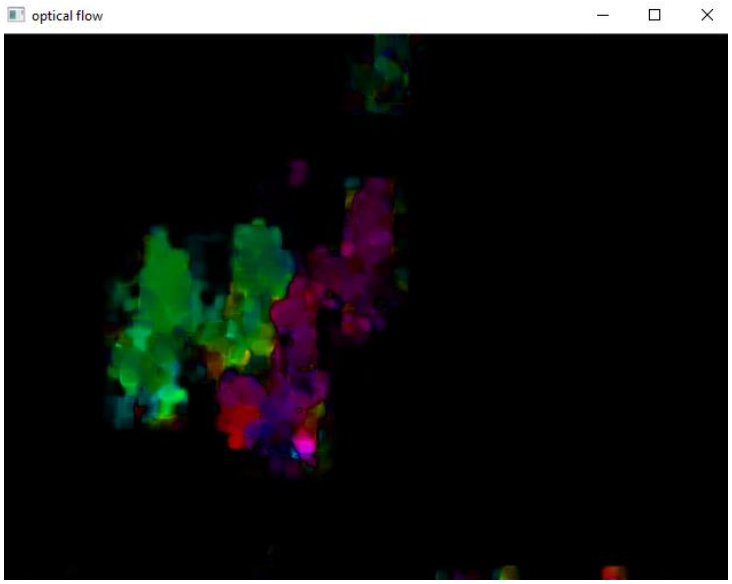
Cilj: Ispitati točnost prebrojavanja kada se promijeni kut snimanja kamere.

Budući da je kod videa 4a kamera nedovoljno fiksirana, tj. dolazi do kretanja čitavog okvira, za ovaj test izabran je video 4b u svrhu dolaženja do jasnijih zaključaka. Takva vrsta snimanja u kojoj se čitav okvir kreće predstavlja nepremostivu prepreku za brojač osoba zasnovan na funkciji optičkog toka jer je kretanje očitano na cijelom okviru i nije moguće nastaviti s daljnjom obradom slike na odgovarajući način.



Sl.4.19. *cv:calcOpticalFlowFarneback i video 4b rezultat*

Na slici 4.19. vidi se nastanak problemi kod ovog kuta postavljanja kamere i metode zasnovane na funkciji optičkog toka. Oni počinju s preklapanjem objekata praćenja unutar okvira. Funkcija optičkog toka nije u mogućosti izolirati piksele koji se kreću iz okvira u okvir prema pripadnosti objektu praćenja.



Sl.4.20. *cv::calOpticalFlowFarneback slike 4.19.*

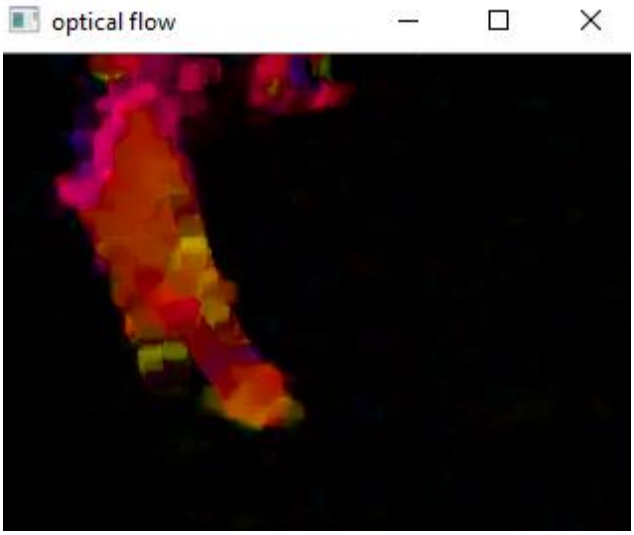
4.2.5. Test video 5

Cilj: Ispitati točnost prebrojavanja na vanjskim mjestima širokog prostora prolaska, niske frekvencije i niske razine promjene svjetlosti pri promjeni kuta kamere

Sl. 4.21. *cv::calcOpticalFlowFarneback i video 5 rezultat*

Budući da je pokazano kako funkcija cv::calcOpticalFlow ima velikih problema u preklapanju objekata praćenja, testiran je ovaj jednostavni video bez mimoilaženja osoba u kadru i točnost se, uz određene probleme, vraća na visoke postotke.

Problem kod ove implementacije rješenja je preklapanje optičkog toka pješaka koji je objekt praćenja i pozadinskog toka kretanja automobila kako se vidi na slici 20. Problem je riješen ograničavanjem prostora slike na kojemu promatramo promjene u optičkom toku između dva okvira.



Sl.4.22. *Preklapanje optičkih tokova pješaka i automobila sa slike 4.21.*

Usporedbom algoritma za brojanje osoba zasnovanog na apsolutnoj razlici između okvira i algoritma za brojanje osoba zasnovanog na optičkom toku okvira uočene su sličnosti i razlike u radu istih. Postotak točnosti prebrojavanja, kao najvažniji parametar, je gotovo identičan za sve video zapise osim video zapisa broj 5 gdje je algoritam zasnovan na apsolutnoj razlici uspješniji sa 100% naspram 75%. Razlog tome su isti problemi koji se pojavljuju kod oba algoritma zbog preklapanja osoba prilikom promjene kuta kamere s vertikalnog na horizontalni. Uočeno je da oba algoritma imaju slabu mogućnost prilagodbe na promjenu kuta kamere pri snimanju, tj. da pri vertikalnom kutu kamere prebrojavaju s visokom točnošću između 96% i 100% dok pri horizontalnom kutu taj postotak pada na svega 50% do 58%. Iznimka su situacije horizontalnog kuta kamere pri kontroliranom prolasku osoba kada se one kreću jedna po jedna, tj. kada nema preklapanja i mimoilaženja. Tada su algoritmi upotrebljivi s točnošću od 75% za optički tok i 100% za apsolutnu razliku između okvira. Osim vertikalnog kuta kamere uočeno je da oba algoritma najtočnije prebrojavaju u situacijama uskih prolaza (vrata). Lokacija u smislu unutarnjih i vanjskih prostora također se nije pokazala kao bitan faktor u uspješnosti rada algoritama, dok je razina osvjetljenja i pojava sjene znatno više utjecala na rad algoritma zasnovanog na optičkom toku. Nadalje, razlike u radu algoritama uočene su u toleranciji na kretanje kamere gdje algoritam zasnovan na optičkom toku pokazuje iznimno niske razine prilagodbe i na najmanje podrhtavanje slike, dok onaj zasnovan na apsolutnoj razlici ima mogućnost tolerancije minimalnog kretanja kamere. Najveća razlika ova dva algoritma je u

vremenu potrebnom za njihovo izvršavanje na osobnom računalu, tj. u korištenju računalnih resursa. Algoritam zasnovan na apsolutnoj razlici između okvira pokazuje se kao brže i bolje rješenje s brzinom izvršavanja na jednom okviru od prosječno 31 ms kada u obzir uzmemo sve video zapise, za razliku od brzine izvršavanja algoritma optičkog toka od prosječno 77 ms po okviru.

4.3. Rezultati implementacije na Raspberry Pi

Raspberry Pi je računalo s Raspbian operacijskim sustavom koji se temelji na Debianu. Iako se pojavljuje u brojnim inačicama za potrebe rada korišten je *Raspberry Pi 3* s operacijskim sustavom Raspbian Jessie – posebno prilagođenim za sklopovlje *Raspberry Pi – a*. Budući da je Debian Linux operacijski sustav, jednostavna prilagodba koda napisanog u programskom okruženju prilagođenom za Windows nije bila moguća zbog razlike u prevoditelju (eng. *compiler*).

Iz gore navedenih razloga programsko rješenje bilo je potrebno prilagoditi Linux operacijskom sustavu ili ga prepisati u drugi programski jezik kojeg openCV biblioteke podržavaju (npr. *Python*). Zbog opsežnosti koda i nedostatka vremena implementacija na *Raspberry Pi* nije izvedena.

Kao što je navedeno na kraju potpoglavlja 4.2., prosječna brzina izvršavanja algoritama rješenja po okviru slike je 31 ms za apsolutnu razliku i 77 ms za optički tok. Prema [17] u kojemu stoji da je za obradu slike pomoću funkcija OpenCV biblioteka *RaspberryPi* računalu trebalo 3.8 puta više vremena nego osobnom računalu može se zaključiti da bi brzina obrade algoritma apsolutne razlike bila 117,8 ms po okviru, a za optički tok 292,6 ms po okviru na toj platformi. Iz ovih podataka proizlazi zaključak da algoritmi u ovom obliku ne bi bili izvedivi u stvarnom vremenu, već bi trebali biti optimizirani za ugradbenu platformu.

5. ZAKLJUČAK

Izrada brojača osoba zasnovanog na kameri tema je ovog diplomskog rada. U radu je najprije dan pregled postojećih načina za brojanje osoba te su predstavljeni nedostatci i prednosti pojedinih načina. Zatim je predloženo rješenje zasnovano na kameri kao senzoru. Osim hardverskog aspekta koji analizira potrebno sklopovlje za razvoj i izradu brojača osoba zasnovanog na kameri, u radu je detaljno predstavljen koncept i implementacija programske podrške za praćenje i brojanje osoba u videu zapisu. U predloženom rješenju detekcija objekata zasniva se na računanju apsolutne razlike između dva uzastopna okvira video zapisa ili na računanju ukupnog optičkog toka okvira.

Tijekom izrade rada oba rješenja su testirana na različite uvjete uključujući lokaciju, razinu osvijetljenosti, položaj kamere, širinu prolaza, tip prometa i frekvenciju prometa. Nakon faze testiranja zaključeno je da oba rješenja postižu vrlo slične rezultate. Također je zaključeno da rješenja pokazuju visoku razinu robusnosti na tip prometa koji je promatran, ali vrlo malu robusnost na promjenu kuta kamere. Najbolji rezultati dobiveni su pri brojanju pod vertikalnim kutom kamere u uskim prolazima.

LITERATURA

[1] https://en.wikipedia.org/wiki/People_counter [Pristupljeno: 12 Rujan 2018.]

[2] https://www.axis.com/ja-jp/products/axis-tailgating-detector [Pristupljeno: 12 Rujan 2018.]

[3] https://docs.opencv.org/3.4.2/d2/de8/group__core__array.html [Pristupljeno: 12 Rujan 2018.]

[4]http://www.opencv.org.cn/opencvdoc/2.3.2/html/modules/video/doc/motion_analysis_and_object_tracking.html [Pristupljeno: 12 Rujan 2018.]

[5] https://docs.opencv.org/3.2.0/db/d8e/tutorial_threshold.html [Pristupljeno: 12 Rujan 2018.]

[6] https://docs.opencv.org/2.4/doc/tutorials/imgproc/shapedescriptors/hull/hull.html [Pristupljeno: 12 Rujan 2018.]

[7] Izvor: http://opencvexamples.blogspot.com/2013/10/convex-hull.html [Pristupljeno: 12 Rujan 2018.]

[8] https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/ [Pristupljeno: 12 Rujan 2018.]

[9] https://opencv.org/about.html [Pristupljeno: 12 Rujan 2018.]

[10] https://visualstudio.microsoft.com/vs/ [Pristupljeno: 12 Rujan 2018.]

[11] https://www.tutorialspoint.com/cplusplus/ [Pristupljeno: 12 Rujan 2018.]

[12] https://www.youtube.com/watch?v=H7BrVzdOzc4 [Pristupljeno 12 Rujan 2018.]

[13] https://www.youtube.com/watch?v=1QPBmihMo0I [Pristupljeno: 12 Rujan 2018.]

[14] https://www.youtube.com/watch?v=RGC7gMFyTpQ [Pristupljeno: 12 Rujan 2018.]

[15] https://www.youtube.com/watch?v=swglFSZdJvI [Pristupljeno: 12 Rujan 2018.]

[16] https://www.youtube.com/watch?v=QetLqQpnfkA [Pristupljeno: 12 Rujan 2018.]

[17] Identifikacija osoba pomoću računalnog vida, Diplomski rad

SAŽETAK

Diplomski rad teme Brojač osoba zasnovan na kameri ima za zadatak razviti i ispitati programsko rješenje brojanja prolaznika u video zapisu pomoću OpenCV biblioteka, C++ programskog jezika i Microsoft Visual Studio programskog okruženja. Za potrebe rada razvijena su dva programska rješenja zasnovana na funkcijama za računanje apsolutne razlike između dva okvira i ukupnog optičkog toka okvira i testirana na pet video zapisa s različitim uvjetima. Analizom rezultata uočene su prednosti i razlike programskih rješenja u danim uvjetima. Budući da postoje brojne verzije brojača osoba zasnovane na različitim tehnologijama, cilj rada je informirati buduće korisnike o ponašanju određene verzije u određenim uvjetima kako bi se skratio put istraživanja pri implementaciji.

Ključne riječi: Brojač, Kamera, OpenCV, C++ Programski jezik, Visual Studio

ABSTRACT

The thesis named Camera-based people counter has the task of developing and examining two software solutions for counting using the OpenCV library, the C ++ programming language, and the Microsoft Visual Studio programming environment. Two software solutions based on the functions cv :: absdiff and cv :: calcOpticalFlowFarneback were developed for the purpose of the thesis and tested on five videos with different conditions. By analyzing the results, the advantages and disadvantages of the program solutions in the given conditions were observed. Since there are numerous versions of people counters based on different technologies, the purpose of the thesis is to inform future users about the behavior of a specific version under certain conditions to shorten the implementation path.

Keywords: People Counter, Camera, OpenCV, C++ Programming language, Visual Studio

ŽIVOTOPIS

Hrvoje Karalić rođen je 3.7.1992. godine u Osijeku. Osnovnu školu „Ivan Filipović“ završio je 2007. godine u Osijeku. Pohađao Prirodoslovno-Matematičku gimnaziju u Osijeku, koju je završio 2011. godine. Iste godine upisao je preddiplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku koji završava 2014. Godine. Trenutno završava drugu godinu diplomskog studija na Fakultetu Elektrotehnike Računarstva i Informacijskih tehnologija u Osijeku.